WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Wednesday, April 28, 2004

Hide?	Set Name	Query	Hit Count
		DB=EPAB,JPAB,DWPI,TDBD; PLUR=NO; OP=ADJ	
☐	L57	(l54 or l55) and L56	2
☐	L56	(forward Compatibil$3) or (backward compatibil$3)	195
☐	L55	(interface$1) with program$5	16672
☐	L54	(interface$1) with configur$4	2522
☐	L53	(old or older) with (coprocessor$1 or co-processor$1)	2
☐	L52	(new or newer) with (coprocessor$1 or co-processor$1)	20
		DB=PGPB,USPT; PLUR=NO; OP=ADJ	
☐	L51	L24 and L49	0
☐	L50	L28 same L49	0
☐	L49	busy with (L12 or L13)	119
☐	L48	L45 and L24	0
☐	L47	L45 and L28	1
☐	L46	L45 same L28	0
☐	L45	L44 same (L12 or L13)	47
☐	L44	(forward Compatibil$3) or (backward compatibil$3)	1900
☐	L43	L42 same (L12 or L13)	7
☐	L42	(new or newer) with (coprocessor$1 or co-processor$1)	297
☐	L41	L40 not L32	18
☐	L40	(L12 or L13) and (L38 or L39)	18
☐	L39	old with (coprocessor$1 or co-processor$1)	26
☐	L38	older with (coprocessor$1 or co-processor$1)	8
☐	L37	obsolete with (coprocessor$1 or co-processor$1)	0
☐	L36	obsolete adj (coprocessor$1 or co-processor$1)	0
☐	L35	older adj (coprocessor$1 or co-processor$1)	0
☐	L34	old adj (coprocessor$1 or co-processor$1)	0
☐	L33	L32 not L30	6
☐	L32	L31 and (L25 or L26)	9
☐	L31	(L12 or L13) same L28	532
☐	L30	L15 and (L25 or L26)	3
☐	L29	L27 and L28	4
☐	L28	coprocessor$1 or co-processor$1	8056

☐	L27	L24 and L25	8
☐	L26	(703/26).ccls.	282
☐	L25	(712/227).ccls.	449
☐	L24	(712/34).ccls.	217
☐	L23	condition code$1 or predicate$1	18489
☐	L22	L6 and L20	81
☐	L21	L6 same L20	2
☐	L20	(co-processor$1 or coprocessor$1)	8056
☐	L19	L6 and L14	2
☐	L18	L17 not L15	51
☐	L17	(L12 or L13) and L16	59
☐	L16	(712/35).ccls.	185
☐	L15	(L12 or L13) and L14	71
☐	L14	(712/34).ccls.	217
☐	L13	(interface$1) with program$5	80732
☐	L12	(interface$1) with configur$4	31748
☐	L11	(L8 or L9) with configur$4	6
☐	L10	(L8 or L9) with program$5	37
☐	L9	co-processor interface	79
☐	L8	coprocessor interface	226
☐	L7	coprocessor with functional units	40
☐	L6	data with L5	2660
☐	L5	L4 adj L3	24643
☐	L4	out	2558170
☐	L3	order	2417518
☐	L2	out of order	0
☐	L1	"out of order"	0

END OF SEARCH HISTORY

WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Wednesday, April 28, 2004

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT; PLUR=NO; OP=ADJ	
☐	L20	l11 with flag$1	95
☐	L19	l11 with L18	3
☐	L18	implicit	15822
☐	L17	l11 same L16	0
☐	L16	implicit operand$1	69
☐	L15	l11 with L14	18
☐	L14	destination register	2813
☐	L13	l11 with index	18
☐	L12	L11 with first with second with destination	4
☐	L11	l6 or l7 or l8 or L10	827
☐	L10	compare macro-instruction$1	1
☐	L9	compare micro-instruction$1	0
☐	L8	compare macroinstruction$1	1
☐	L7	compare microinstruction$1	6
☐	L6	compare instruction$1	822
☐	L5	l1 and base address	1
☐	L4	802196.ap.	2
☐	L3	r7 and L2	0
☐	L2	vliw and L1	1
☐	L1	6542990.pn.	1

END OF SEARCH HISTORY

First Hit

Generate Collection Print

L52: Entry 5 of 20 File: EPAB Mar 18, 1992

PUB-NO: EP000475028A2
DOCUMENT-IDENTIFIER: EP 475028 A2
TITLE: Procedure for operating a coprocessor in a distributed computer system.

PUBN-DATE: March 18, 1992

INVENTOR-INFORMATION:

NAME	COUNTRY
GEIGER, MICHAEL	DE
JENSEN, THOMAS-HERLIN	DK
ZOLG, MARKUS	DE

ASSIGNEE-INFORMATION:

NAME	COUNTRY
SIEMENS AG	DE

APPL-NO: EP91112265
APPL-DATE: July 22, 1991

PRIORITY-DATA: DE04027324A (August 29, 1990)

INT-CL (IPC): G06F 9/38
EUR-CL (EPC): G06F009/38; G06F009/38

ABSTRACT:

CHG DATE=19990617 STATUS=O> In the operation of a coprocessor in a distributed computer system, instructions to be executed by the coprocessor are communicated to it by a computer. The coprocessor is connected to the computer via an interface unit. A software unit having three functional units is contained in this interface unit. The first functional unit receives the messages from the computer in which the instruction to be executed is contained with necessary parameters, and unpacks this message and decodes the instruction. In accordance with the decoded instruction, the second functional unit supplies the function, which can access the coprocessor. If this function is, for example, a memory access to the memory of the coprocessor, this is initiated directly by the second functional unit. If the function to be executed is a processing step which is to be executed by the coprocessor, this is communicated to a third functional unit, which communicates the function to be executed to the coprocessor with the aid of an interrupt. The coprocessor then processes the instruction and communicates completion of the task by a further interrupt. By means of the second functional unit, this is communicated to the first functional unit and at the same time the result data are transferred to the functional unit. The first functional unit uses the results to put together again a computer-specific message, which is transferred to the computer. The concept of the software unit is such that adaptations to new coprocessors or to a new modified operating system of the computer system require only slight modifications of the software unit.

First Hit

Generate Collection Print

L52: Entry 6 of 20 File: TDBD Oct 1, 1996

TDB-ACC-NO: NN9610151

DISCLOSURE TITLE: Developing Embedded System Control Programs

PUBLICATION-DATA:
IBM Technical Disclosure Bulletin, October 1996, US

VOLUME NUMBER: 39
ISSUE NUMBER: 10
PAGE NUMBER: 151 - 152

PUBLICATION-DATE: October 1, 1996 (19961001)

CROSS REFERENCE: 0018-8689-39-10-151

DISCLOSURE TEXT:

Many modern complex computer systems use common industry standard UNIX* operating systems and workstations for their non-real-time components and specialized coprocessors for their real-time components. The UNIX host is typically used to present an interface to human operators and for batch operations such as database lookup while the coprocessors are typically used for the time-critical movement of communications data or for controlling complex hardware.

The run-time operation of such systems usually use carefully designed protocols of communication between processes on the UNIX host and processes running on the coprocessor. Designing such protocols is a painstaking and time-consuming business and some phases of operation of such systems---such as initialization and the diagnostic phases---would better be served by an approach that allowed for the rapid development of software to control the embedded system.

The approach described permits the development of software for controlling embedded software that can be used during initial development through the systems live operation and during diagnosis when problems are being investigated. There is first developed, for a given coprocessor, a pair of communicating processes that provide access for processes on the coprocessor to the standard file input/output facilities of the UNIX host. One process resides on the UNIX host and thus has access to its file system. The other process is implemented as a library that is linked with coprocessor tasks and provides a set of routines that give access to the UNIX file facilities. When one of the library of routines is called, the library process uses a communication protocol to cooperate with the UNIX process to access UNIX files. The set of routines supported should be: 1. printf function to allow parameterized string data to be written to the standard output stream of the UNIX process. 2. gets function to allow the coprocessor task to input a line of string data from the UNIX process' standard input stream. In addition, routines that allowed a coprocessor process to download the data contained in a UNIX file into coprocessor memory and a routine that allowed the process to dump a block of memory into a UNIX file would be most useful. A framework for using the coprocessor is thus developed whenever a new coprocessor is picked for a development project. Once this framework has been developed, then processes can be quickly developed to run on the coprocessor independent of the communication mechanism used with the

UNIX host. Instead, coprocessor processes can be written to use simple ASCII menu driven interfaces for their operation. Thus, during the development phase of the project, software developed in this way can be manually operated by developers through an interactive prototyping cycle.

Once a coprocessor process has been developed, its operation may be automated by the use of UNIX korn shell scripts. If a korn shell script is written to start the UNIX process that cooperates with the embedded process as a coprocess, then the script can direct commands to the standard input stream of the UNIX process and can parse the results of the embedded process that are directed to the standard output stream of the UNIX process. Thus, during the live operation of the system, the coprocess task is automatically automated by the shell scripts.

If a problem is discovered with a system installed in the field, then a customer engineer can run the same coprocess tasks manually without their korn shell wrappers to attempt to diagnose the problem. This approach accordingly allows developers to rapidly develop systems for controlling embedded software and permits the same software to be used during the startup, running and support phases of a project. * Trademark in the United States and other countries licensed exclusively through X/Open Company Limited

First Hit

Generate Collection Print

L52: Entry 11 of 20 File: DWPI Nov 27, 2003

DERWENT-ACC-NO: 2004-022953
DERWENT-WEEK: 200402

TITLE: Microprocessor instruction extension adding system, has enhanced extension language for capturing new instructions and description of load/stored instructions executed by corresponding logic shared with core instructions

INVENTOR: GONZALES, R E; KILLIAN, E A ; WANG, A ; WILSON, R P

PATENT-ASSIGNEE:

ASSIGNEE	CODE
TENSILICA INC	TENSN

PRIORITY-DATA: 2002US-0146655 (May 13, 2002)

Search Selected Search ALL Clear

PATENT-FAMILY:

	PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
☐	WO 2003098379 A2	November 27, 2003	E	059	G06F000/00

DESIGNATED-STATES: AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ OM PH PL PT RO RU SC SD SE SG SK SL TJ TM TN TR TT TZ UA UG UZ VC VN YU ZA ZM ZW AT BE BG CH CY CZ DE DK EA EE ES FI FR GB GH GM GR HU IE IT KE LS LU MC MW MZ NL OA PT RO SD SE SI SK SL SZ TR TZ UG ZM ZW

APPLICATION-DATA:

PUB-NO	APPL-DATE	APPL-NO	DESCRIPTOR
WO2003098379A2	April 14, 2003	2003WO-US11639	

INT-CL (IPC): G06 F 0/00

ABSTRACTED-PUB-NO: WO2003098379A
BASIC-ABSTRACT:

NOVELTY - The system has a tensilica extension language (TIE) for formal capturing of both new instructions for execution by VLIW co-processor (208). The extension language enables description of complex load/store instructions for use by a configurable number of load/store units (212). The description of load/stored instructions is executed by corresponding logic that is shared with core (104) instructions in microprocessor.

USE - Used for adding advanced instruction extensions to a microprocessor.

ADVANTAGE - The system generates fully pipelined micro-architectural implementation for the new instruction in the form of synthesizable HDL description, which can be processed by standard CAD tools. The system allows the designers to design a VLIW microprocessor customized for a specific application to achieve higher performance, lower hardware cost and lower power consumption.

DESCRIPTION OF DRAWING(S) - The drawing shows a block diagram of a high performance processor.

Core 104

Share functions 206

VLIW co- processor 208

Fetch unit 210

Load/store units 212

Register file 214

CHOSEN-DRAWING: Dwg.2/19

TITLE-TERMS: MICROPROCESSOR INSTRUCTION EXTEND ADD SYSTEM ENHANCE EXTEND LANGUAGE CAPTURE NEW INSTRUCTION DESCRIBE LOAD STORAGE INSTRUCTION EXECUTE CORRESPOND LOGIC SHARE CORE INSTRUCTION

DERWENT-CLASS: T01

EPI-CODES: T01-F01B; T01-F03;

SECONDARY-ACC-NO:
Non-CPI Secondary Accession Numbers: N2004-017725

First Hit

Generate Collection Print

L52: Entry 14 of 20 File: DWPI Jul 13, 1999

DERWENT-ACC-NO: 1999-404631
DERWENT-WEEK: 199934

TITLE: Interfacing method between processor and coprocessor in data processing systems

INVENTOR: ARENDS, J; MOYER, W C ; SCOTT, J W

PATENT-ASSIGNEE: MOTOROLA INC (MOTI)

PRIORITY-DATA: 1997US-0924137 (September 5, 1997)

Search Selected Search ALL Clear

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
US 5923893 A	July 13, 1999		033	G06F015/76

APPLICATION-DATA:

PUB-NO	APPL-DATE	APPL-NO	DESCRIPTOR
US 5923893A	September 5, 1997	1997US-0924137	

INT-CL (IPC): G06 F 15/76

ABSTRACTED-PUB-NO: US 5923893A
BASIC-ABSTRACT:

NOVELTY - A processor (12) receives the instruction and decodes, which is indicated to a coprocessor (14) by providing a control signal through a one part of a coprocessor bus (28). At least one information is transferred from the processor to the coprocessor through the other part of the bus. The bus is indicated of the transfer by providing another control signal.

USE - For interfacing processor to coprocessor like math coprocessor, multiply-accumulators, modems, digital signal processors, vitturbi calculators, cryptographic processors, image processors and vector processors.

ADVANTAGE - Abstracts and isolates the operation of the coprocessor from the primary processor, lessening the effort required to integrate a new coprocessor with an existing processor. Makes the interface programmer friendly in order to facilitate tailoring new coprocessor applications in software instead of in hardware.

DESCRIPTION OF DRAWING(S) - The figure illustrates the data processing system with processor being interfaced to the coprocessor.

Processor 12

Coprocessor 14

Coprocessor bus 28

ABSTRACTED-PUB-NO: US 5923893A
EQUIVALENT-ABSTRACTS:

CHOSEN-DRAWING: Dwg.1/26

DERWENT-CLASS: T01
EPI-CODES: T01-M;

WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Wednesday, April 28, 2004

Hide?	Set Name	Query	Hit Count
		DB=EPAB,JPAB,DWPI,TDBD; PLUR=NO; OP=ADJ	
☐	L57	(l54 or l55) and L56	2
☐	L56	(forward Compatibil$3) or (backward compatibil$3)	195
☐	L55	(interface$1) with program$5	16672
☐	L54	(interface$1) with configur$4	2522
☐	L53	(old or older) with (coprocessor$1 or co-processor$1)	2
☐	L52	(new or newer) with (coprocessor$1 or co-processor$1)	20
		DB=PGPB,USPT; PLUR=NO; OP=ADJ	
☐	L51	L24 and L49	0
☐	L50	L28 same L49	0
☐	L49	busy with (L12 or L13)	119
☐	L48	L45 and L24	0
☐	L47	L45 and L28	1
☐	L46	L45 same L28	0
☐	L45	L44 same (L12 or L13)	47
☐	L44	(forward Compatibil$3) or (backward compatibil$3)	1900
☐	L43	L42 same (L12 or L13)	7
☐	L42	(new or newer) with (coprocessor$1 or co-processor$1)	297
☐	L41	L40 not L32	18
☐	L40	(L12 or L13) and (L38 or L39)	18
☐	L39	old with (coprocessor$1 or co-processor$1)	26
☐	L38	older with (coprocessor$1 or co-processor$1)	8
☐	L37	obsolete with (coprocessor$1 or co-processor$1)	0
☐	L36	obsolete adj (coprocessor$1 or co-processor$1)	0
☐	L35	older adj (coprocessor$1 or co-processor$1)	0
☐	L34	old adj (coprocessor$1 or co-processor$1)	0
☐	L33	L32 not L30	6
☐	L32	L31 and (L25 or L26)	9
☐	L31	(L12 or L13) same L28	532
☐	L30	L15 and (L25 or L26)	3
☐	L29	L27 and L28	4
☐	L28	coprocessor$1 or co-processor$1	8056

☐	L27	L24 and L25	8
☐	L26	(703/26).ccls.	282
☐	L25	(712/227).ccls.	449
☐	L24	(712/34).ccls.	217
☐	L23	condition code$1 or predicate$1	18489
☐	L22	L6 and L20	81
☐	L21	L6 same L20	2
☐	L20	(co-processor$1 or coprocessor$1)	8056
☐	L19	L6 and L14	2
☐	L18	L17 not L15	51
☐	L17	(L12 or L13) and L16	59
☐	L16	(712/35).ccls.	185
☐	L15	(L12 or L13) and L14	71
☐	L14	(712/34).ccls.	217
☐	L13	(interface$1) with program$5	80732
☐	L12	(interface$1) with configur$4	31748
☐	L11	(L8 or L9) with configur$4	6
☐	L10	(L8 or L9) with program$5	37
☐	L9	co-processor interface	79
☐	L8	coprocessor interface	226
☐	L7	coprocessor with functional units	40
☐	L6	data with L5	2660
☐	L5	L4 adj L3	24643
☐	L4	out	2558170
☐	L3	order	2417518
☐	L2	out of order	0
☐	L1	"out of order"	0

END OF SEARCH HISTORY

WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Wednesday, April 28, 2004

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT; PLUR=NO; OP=ADJ	
☐	L51	L24 and L49	0
☐	L50	L28 same L49	0
☐	L49	busy with (L12 or L13)	119
☐	L48	L45 and L24	0
☐	L47	L45 and L28	1
☐	L46	L45 same L28	0
☐	L45	L44 same (L12 or L13)	47
☐	L44	(forward Compatibil$3) or (backward compatibil$3)	1900
☐	L43	L42 same (L12 or L13)	7
☐	L42	(new or newer) with (coprocessor$1 or co-processor$1)	297
☐	L41	L40 not L32	18
☐	L40	(L12 or L13) and (L38 or L39)	18
☐	L39	old with (coprocessor$1 or co-processor$1)	26
☐	L38	older with (coprocessor$1 or co-processor$1)	8
☐	L37	obsolete with (coprocessor$1 or co-processor$1)	0
☐	L36	obsolete adj (coprocessor$1 or co-processor$1)	0
☐	L35	older adj (coprocessor$1 or co-processor$1)	0
☐	L34	old adj (coprocessor$1 or co-processor$1)	0
☐	L33	L32 not L30	6
☐	L32	L31 and (L25 or L26)	9
☐	L31	(L12 or L13) same L28	532
☐	L30	L15 and (L25 or L26)	3
☐	L29	L27 and L28	4
☐	L28	coprocessor$1 or co-processor$1	8056
☐	L27	L24 and L25	8
☐	L26	(703/26).ccls.	282
☐	L25	(712/227).ccls.	449
☐	L24	(712/34).ccls.	217
☐	L23	condition code$1 or predicate$1	18489
☐	L22	L6 and L20	81
☐	L21	L6 same L20	2

☐	L20	(co-processor$1 or coprocessor$1)	8056
☐	L19	L6 and L14	2
☐	L18	L17 not L15	51
☐	L17	(L12 or L13) and L16	59
☐	L16	(712/35).ccls.	185
☐	L15	(L12 or L13) and L14	71
☐	L14	(712/34).ccls.	217
☐	L13	(interface$1) with program$5	80732
☐	L12	(interface$1) with configur$4	31748
☐	L11	(L8 or L9) with configur$4	6
☐	L10	(L8 or L9) with program$5	37
☐	L9	co-processor interface	79
☐	L8	coprocessor interface	226
☐	L7	coprocessor with functional units	40
☐	L6	data with L5	2660
☐	L5	L4 adj L3	24643
☐	L4	out	2558170
☐	L3	order	2417518
☐	L2	out of order	0
☐	L1	"out of order"	0

END OF SEARCH HISTORY